

AD-A086 564

NAVAL POSTGRADUATE SCHOOL MONTEREY CA

F/8 15/7

SOME UTILITY PROGRAMS FOR THE STAR COMBAT SIMULATION MODEL. (U)

JUN 80 J K HARTMAN

MIPR-CD-1-80

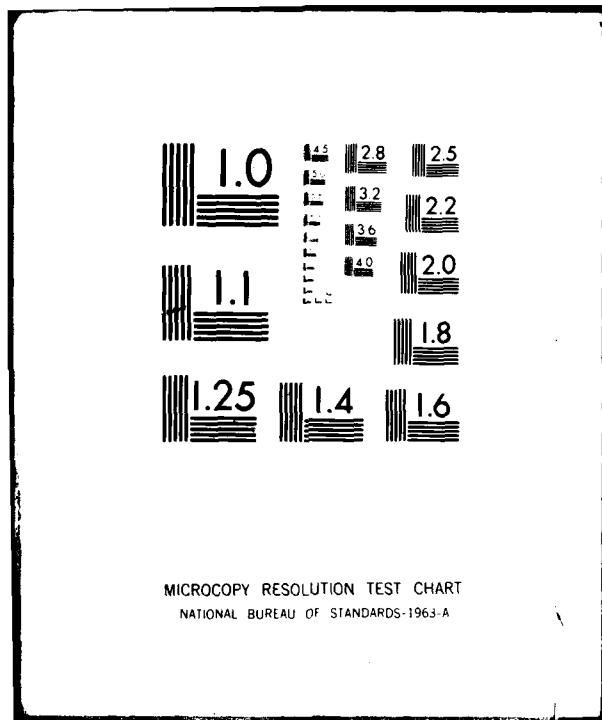
UNCLASSIFIED

NPS55-80-022

NL

100-1
N
Schiff

END
DATE
8-80
DTIC



ADA 086564

LEVEL

2
A

NPS55-80- 022

NAVAL POSTGRADUATE SCHOOL
Monterey, California



JUL 16 1980

A

SOME UTILITY PROGRAMS FOR THE
~~COMPUTER~~ STAR SIMULATION MODEL

by

James K. HARTMAN

June 1980

Approved for public release; distribution unlimited.

Prepared for:

The U.S. Army Training & Doctrine Command
Fort Monroe, VA.

DDC FILE COPY

80 7 14 098

NAVAL POSTGRADUATE SCHOOL
Monterey, California

Rear Admiral J. J. Ekelund
Superintendent

Jack R. Borsting
Provost

The work reported herein was accomplished with the support
of the U.S. Army Training & Doctrine Command, Fort Monroe, VA.

Reproduction of all or part of this report is authorized.

Prepared by:

James K. Hartman
James K. Hartman
Department of Operations Research

Reviewed by:

Michael G. Sovereign
Michael G. Sovereign, Chairman
Department of Operations Research

Released by:

William M. Tolles
William M. Tolles
Dean of Research

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

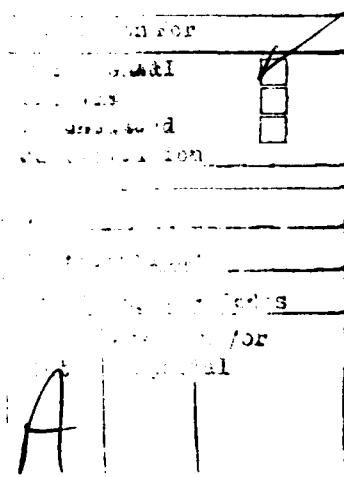
REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER <i>(14)</i> NPS55-80-022	2. GOVT ACCESSION NO. <i>AD-A086 564</i>	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) <i>(6)</i> Some Utility Programs for the STAR Combat Simulation Model	5. TYPE OF REPORT & PERIOD COVERED <i>(7)</i> Technical Report	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) <i>(10)</i> James K. Hartman	8. CONTRACT OR GRANT NUMBER(s)	
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS <i>(15)</i> MIPR-CD-1-80	
11. CONTROLLING OFFICE NAME AND ADDRESS U.S. Army TRADOC Fort Monroe, VA 23651	12. REPORT DATE <i>(11)</i> June 1980	13. NUMBER OF PAGES <i>(22)</i>
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) <i>(12 26)</i>	15. SECURITY CLASS. (of this report) UNCLASSIFIED	16a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) STAR Combat models Contour map Plotter		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This report is a user's manual for several utility programs which have been written to aid the program development effort for the STAR combined arms combat simulation model.		

251450

JP

TABLE OF CONTENTS

I. Introduction	1
II. Battlefield Map Plot Programs	2
A. Plot Options	2
B. Preprocessor Programs and Disk Data Files	9
III. Code Reconciliation Program - RECON	12
A. Introduction	12
B. Inputs	12
C. Module Definition and Sequencing	13
D. Comparisons Inside a Module	15
IV. Code Plot Program	17



I. INTRODUCTION

This report provides a user's manual for several utility programs, which have been used as aids during the development of the STAR combat simulation model.

Section II describes a program for rapidly producing contour maps of the STAR battlefield area and overlaying information of various types on the maps.

Section III describes a program for reconciling two computer programs which are mostly identical, but contain some differences. Such programs have frequently resulted during the STAR development as several groups simultaneously build new modules for incorporation into the base program.

Section IV describes a program for producing Tech report or thesis - ready copies of computer code without requiring photo-reduction.

II. BATTLEFIELD MAP PLOT PROGRAMS

As an aid to visualizing the progress of a simulated battle, and especially as an aid in the scenario formulation process, a program has been written to rapidly prepare contour maps of the STAR battlefield area with a series of optional overlays. The program is written in FORTRAN and uses standard VERSATEC plotter routines. [ref. 1] Several data sets defining the maps are stored on disk by preprocessor programs and read back by the plot program. This saves substantial time since the preprocessing is quite lengthy for preparing the contour map. Section A of this chapter will describe the plot program options. Section B briefly discusses the preprocessor programs.

A. Plot Options

A program listing for the map plot program is given in Figure 1. This particular program is set up to plot a 10 x 10 kilometer map segment at a scale of 1:25000 yielding a map 40 x 40 cm. (another version of the program yields a 30 x 10 km. map area). Plot units for the program are meters, so map coordinates to be plotted are in the range 0. to 10,000. for both X and Y coordinates. The program currently has 9 active options which are selected by the first input data card: a 1 in column j selects option j and a 0 in column j suppresses the option (j=1,...,9).

1. Option 1 Coordinate Grid. This option, if selected, plots a 1 x 1 km grid over the map area.
2. Option 2 Label Coordinates. This option prints UTM coordinates for each kilometer in the map boundary.
3. Option 3 Terrain Contour Map. Option 3 plots a contour map of the battlefield area. Substantial preprocessing (see section B) is done to prepare and place on a disk file the X, Y coordinates defining each contour

```

1 REPLACE THIS CARD WITH STANDARD JOB CARD WITH TIME = 5
2 // EXEC FORTCLGM
3 //FORT.SYSIN DD *
4 C PLOT PROGRAM FOR 10 X 10 KM TERRAIN BOX.
5      DIMENSION IOPT(10),BX(7),BY(7),EX(7),EY(7),X(3600),Y(3600),
6      1 1TR(100),ITITLE(20)
7      DATA BX/-500.,10500.,10500.,-500.,-500., 0.,1./,
8      1  BY/-500.,-500.,10500.,10500.,-500., 0.,1./,
9      2  EX/0.,10000.,10000.,0.,0.,0.,1./,
10     3  EY/0.,0.,10000.,10000.,0.,0.,1./
11     DATA LMASK1/Z0FOF/
12
13 C INPUT OPTIONS AS 1 = DESIRED, 0 = NOT DESIRED
14 C      IOPT(1) -- COORDINATE GRID
15 C      (2) -- LABEL COORDINATES
16 C      (3) -- TERRAIN CONTOUR MAP
17 C      (4) -- ACCENT CONTOURS DIVISIBLE BY 100.
18 C      (5) -- FORESTS SHADED
19 C      (6) -- DRAW LINES (EG. ROUTES)
20 C      (7) -- DRAW SYMBOLS (EG. POSITIONS)
21 C      (8) -- TITLE
22 C      (9) -- OUTLINE ELLIPSES (EG. FIELDS)
23 C
24      READ(5,7) IOPT
25      7  FORMAT(10I1)
26 C BATTLEFIELD LOWER LEFT CORNER COORDS IN METERS
27      XLOBY=50000.
28      YLOBY=93000.
29 C
30 C PLOT FRAME
31 C
32      CALL PLOTS(0,0,0)
33      CALL NEWPEN(5)
34      CALL LINE(BX,BY,5,1,0,0)
35      CALL LINE(EX,EY,5,1,0,0)
36      CALL NEWPEN(1)
37 C
38 C PLOT COORDINATE GRID
39 C
40      IF (IOPT(1).NE.1) GO TO 200
41      WRITE(6,107)
42      107 FORMAT(' OPTION 1 -- COORDINATE GRID')
43      CALL GRID(0.,0.,10,1000.,10,1000.,LMASK1)
44 C
45 C PLOT COORDINATE LABEL NUMBERS
46 C
47      200 IF (IOPT(2).NE.1) GO TO 300
48      WRITE(6,207)
49      207 FORMAT(' OPTION 2 -- COORDINATE LABELS')
50      CX=-100.

```

Figure 1. Map Plot Program

```

51      CY=-315.
52      CYT=10185.
53      DX=-445.
54      DXT=10055.
55      DY=-85.
56      XB=XLBY/1000.
57      YB=YLBY/1000.
58      HT = 130.
59      DO 250 I=1,11
60      CALL NUMBER(CX,CY,HT,XB,0.0,-1)
61      CALL NUMBER(CX,CYT,HT,XB,0.0,-1)
62      CX=CX+1000.
63      XB=XB+1
64      CALL NUMBER(DX,DY,HT,YB,0.0,-1)
65      CALL NUMBER(DXT,DY,HT,YB,0.0,-1)
66      DY=DY+1000.
67      250  YB=YB+1
68      C
69      C PLOT TERRAIN CONTOUR MAP
70      C
71      300  IF (IOPT(9).NE.1) GO TO 500
72      WRITE (6,307)
73      307  FORMAT(' OPTION 3 -- TERRAIN CONTOUR LINES')
74      IF (IOPT(4).EQ.1) WRITE (6,407)
75      407  FORMAT(' OPTION 4 -- ACCENT 100 M. CONTOURS')
76      310  READ (3,317,END=390) NP,CV
77      317  FORMAT (1S,F10.0)
78      READ (3,327) (X(I),Y(I),I=1,NP)
79      327  FORMAT (8F10.2)
80      X(NP+1)=0.
81      X(NP+2)=1.
82      Y(NP+1)=0.
83      Y(NP+2)=1.
84      CALL NEWPEN(1)
85      IF (IOPT(4).NE.1) GO TO 350
86      C
87      C ACCENT CONTOURS DIVISIBLE BY 100.
88      C
89      ICV=CV/100.
90      XXCV=CV-ICV*100.
91      IF (ABS(XXCV).LT.0.1) CALL NEWPEN(4)
92      350  CALL LINE(Y,X,NP,1,0,0)
93      GO TO 310
94      C OUT OF DATA
95      390  CALL NEWPEN(1)
96      C
97      C SHADE FORESTED AREAS
98      C
99      500  IF (IOPT(5).NE.1) GO TO 600
100     WRITE (6,517)

```

Figure 1 (Continued).

```

101      517  FORMAT(' OPTION 5 -- SHADE FORESTS')
102      HT=75.
103      XC=50.
104      DO 570 I=1,100
105      READ(2,507) (ITR(J),J=1,100)
106      507  FORMAT(50I1,30X)
107      YC=50.
108      DO 540 J=1,100
109      IF (ITR(J).EQ.0) GO TO 520
110      CALL SYMBOL(XC,YC,HT, 9.0.,-1)
111      520  YC=YC+100.
112      540  CONTINUE
113      XC=XC+100.
114      570  CONTINUE
115      C
116      C PLOT LINES (EG. ROUTES)
117      C
118      600  IF (IOPT(6).NE.1) GO TO 700
119      WRITE(6,627)
120      627  FORMAT(' OPTION 6 -- PLOT LINES')
121      CALL NEWPEN(2)
122      610  READ(5,607) NP
123      607  FORMAT(1I5)
124      IF (NP.EQ.999) GO TO 690
125      WRITE(6,607) NP
126      READ(5,617) (X(I),Y(I),I=1,NP)
127      WRITE(6,617) (X(I),Y(I),I=1,NP)
128      617  FORMAT(8F10.0)
129      DO 640 I=1,NP
130      X(I) = X(I) - XLOBY
131      640  Y(I) = Y(I) - YLOBY
132      X(NP+1)=0.
133      X(NP+2)=1.
134      Y(NP+1)=0.
135      Y(NP+2)=1.
136      CALL LINE(X,Y,NP,1,0,0)
137      GO TO 610
138      690  CALL NEWPEN(1)
139      C
140      C PLOT SYMBOLS (EG. POSITIONS)
141      C
142      700  IF (IOPT(7).NE.1) GO TO 800
143      WRITE(6,707)
144      707  FORMAT(' OPTION 7 -- PLOT POSITIONS')
145      HT = 50.
146      710  READ(5,717) XC,YC,ISYM
147      717  FORMAT(2F10.0,1I5)
148      IF (ISYM.EQ.999) GO TO 800
149      XC=XC-XLOBY
150      YC=YC-YLOBY

```

Figure 1 (Continued).

```

151      CALL SYMBOL(XC,YC,HT, 1STH,0.,-1)
152      GO TO 710
153      C
154      C PLOT TITLE
155      C
156      800  IF (IOP(8).NE.1) GO TO 900
157      READ (5,807) ITITLE
158      807  FORMAT(20A4)
159      WRITE(6,817) ITITLE
160      817  FORMAT(' OPTION 8 -- TITLE  ',20A4)
161      XC=-700.
162      YC=-400.
163      HT=135.
164      CALL SYMBOL(XC,YC,HT,ITITLE,90,0,80)
165      C
166      C OUTLINE ELLIPSES
167      C
168      900  IF (IOP(9) .NE.1) GO TO 1000
169      WRITE(6,907)
170      907  FORMAT(' OPTION 9 -- PLOT ELLIPSES')
171      910  READ(5,917) NUM,XC,YC,SAMAJ,SAMIN,ANGLE
172      WRITE(6,917) NUM,XC,YC,SAMAJ,SAMIN,ANGLE
173      917  FORMAT(15,5F10.2)
174      IF (NUM .EQ. 999) GO TO 1000
175      CALL NEWPEN(1)
176      XC=XC-XLCBY
177      YC=YC-YLCBY
178      TWOPi=3.14159265*2.0
179      ANGLE=ANGLE * TWOPi / 360.
180      ASQ= SAMAJ**2
181      BSQ=SAMIN**2
182      C=TWOPi * SQRT((ASQ+BSQ)/2.0)
183      ZN=C/50.0
184      N=ZN+1
185      NC=-1
186      THETA=0.0
187      DO 950 I=1,N
188      RSQ=(ASQ+BSQ) / (ASQ*(SIN(THETA)**2)+BSQ*(COS(THETA)**2))
189      R=SQRT(RSQ)
190      APT=ANGLE + THETA
191      XX=XC+R*COS(APT)
192      YY=YC+R*SIN(APT)
193      CALL SYMBOL(XX,YY,25.,1,0.,NC)
194      NC=-2
195      THETA=THETA + 50.0/R
196      950  CONTINUE
197      GO TO 910
198      1000 CONTINUE
199      CALL PLOT(0.,0.,999)
200      STOP

```

Figure 1 (Continued).

```
201      END
202      REPLACE THIS CARD WITH A STANDARD ORANGE END OF FILE CARD
203      //GO.PLOTPARM DD *
204      *PLOT XMIN=-999.,XMAX=12000.,YMIN=-999.,YMAX=12000.,UNITS=.0254,
205      SCALE=.00004,STRIP=14000.  *END
206      //GO.FT02F001 DD UNIT=2314,VOL=SER=PA1001,DSN=PLTFLTR,DISP=SHR
207      //GO.FT03F001 DD UNIT=2314,VOL=SER=PA1001,DSN=PLTFL50,DISP=SHR
208      //GO.SYSIN DD *
209      1111111110
210      3
211      52300.    96150.    51620.    96350.    50950.    93100.
212      999
213      52350.    96200.      14
214      52400.    96200.      00
215      52450.    96300.      14
216      0.0      0.0      999
217      TEST PLOT FOR TECH REPORT
218      1 51500.    95000.    500.      200.      45.
219      999
220      REPLACE THIS CARD WITH A STANDARD ORANGE END OF FILE CARD
```

Figure 1 (Continued).

line on the map. This file, which contains perhaps 50,000 X, Y pairs is read by the plot program and plotted. The current file uses 10 m. contour spacing.

4. Option 4. Accent Contours. If this option is selected along with option 3, then contour lines corresponding to elevations evenly divisible by 100 will be plotted darker than the others.

5. Option 5. Shade Forests. The plot program reads from a disk file a 100m. grid of 0-1 values over the entire battlefield. When a 1 is read, signifying the presence of forests in that 100 x 100 grid, a Y symbol is plotted on the map.

6. Option 6. Draw Lines. If option 6 is selected, the plot program will read X, Y coordinates from cards, and connect them by straight lines. This option is useful for plotting routes for moving vehicles or firer/target pairs for each shot. The data should be punched as follows:

Card 1 NP - number of points to follow (Format I5)

Card 2 X₁ Y₁ X₂ Y₂ etc. for NP points (Format 8F10.0)

: continuing on successive cards as needed.

Card k NP - for the next line to be plotted

Card k+1 X, Y, X, Y coordinates for the line

: continuing on successive cards as needed.

Card n NP = 999 signal to end option 6.

7. Option 7 Plot Symbols. This option reads data from cards and plots the indicated symbols on the map. The option is useful for plotting snapshots of element positions during the simulated battle. One card is read for each symbol to be plotted. The card contains three values: X, Y, (the X and Y coordinates of the symbol location on the map) and ISYM (an integer code for the symbol to plot) read in format (2F10.0, I5). The list of available

symbol codes is given in the VERSATEC Graphics Manual, reference [1], page B-4. A symbol value ISYM = 999 terminates the option.

8. Option 8 Print Title. Option 8 reads a single card of character data and plots it on the left hand border of the map.

9. Option 9 Plot Ellipses. Option 9 reads from cards the parameters of ellipses which are then plotted overlaying the battlefield. This option is useful for showing the location of minefields and other areas on the battlefield. One card is read for each ellipse containing the following values:

NUM - sequence number - (999 for end of data) (I5)

XC - X coordinate of ellipse center (F10.2)

YC - Y coordinate of ellipse center (F10.2)

SAMAJ - length of semi-major axis (F10.2)

SAMIN - length of semi-minor axis (F10.2)

ANGLE - angle in degrees measured counterclockwise from east to the major axis (F10.2)

A section from a map plotted by this program is included as Figure 2. All of the options were used in its creation. The data input cards for this plot are shown in Figure 1.

B. Preprocessor Programs and Disk Data Files

The data used to plot the contour lines and forested areas is computed in advance and stored on disk to speed execution of the plot program.

The forest preprocessor does a simple 100m. scan of the battlefield area and evaluates tree height at each sample point. Tree height of zero is stored as a 0, and does not plot a symbol at that point. Positive tree height

TEST PLOT FOR TECH REPORT

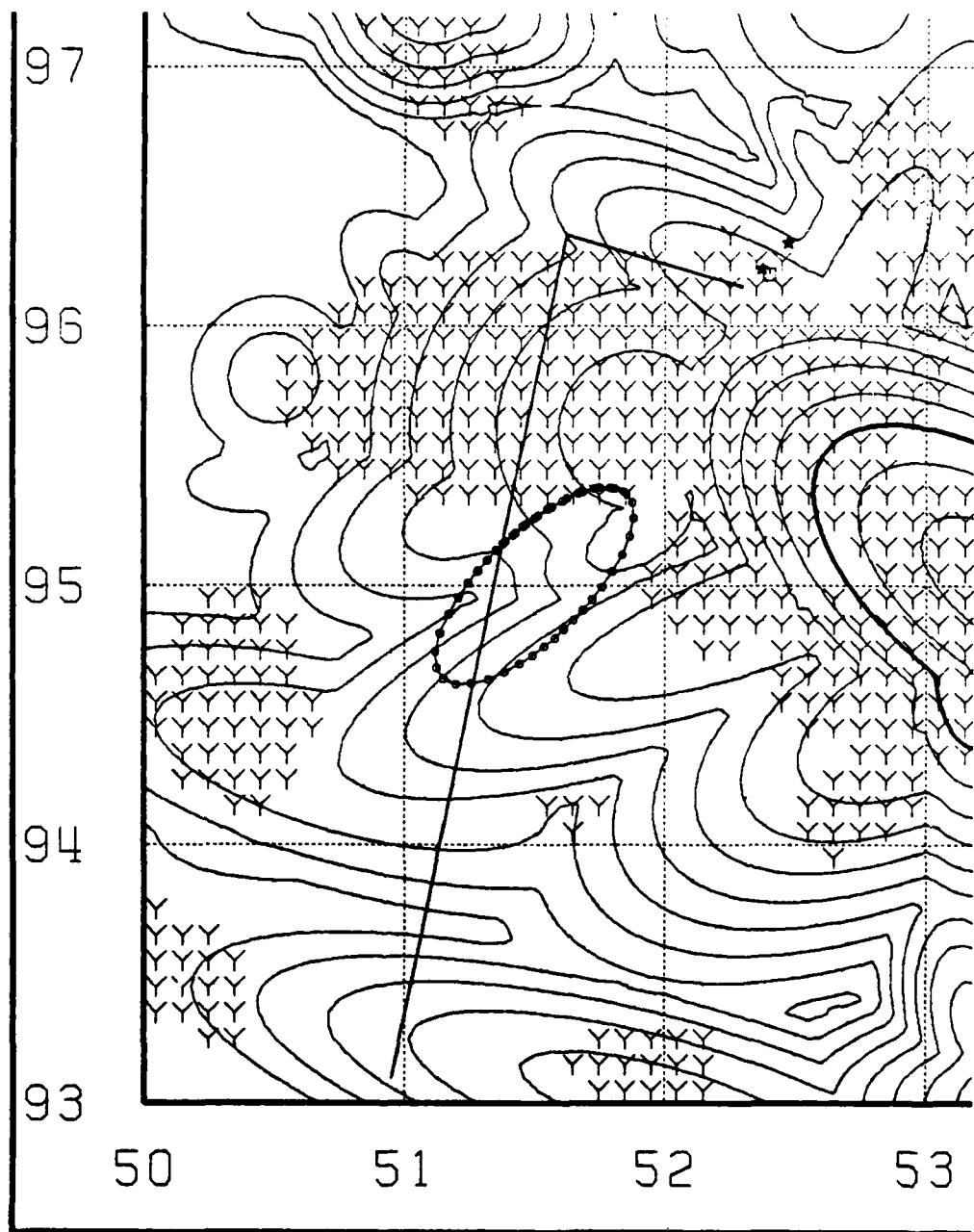


Figure 2. Sample Map

results in a 1 stored on disk and a Y symbol plotted. The tree data is read from FORTRAN data set FT02F001 by the plot program.

The contour map preprocessor is much more complex, and was borrowed from the NPS Computer Center's library contour mapping routine CONTUR. The routine was modified so that instead of generating plot calls directly it now writes X,Y line coordinates to disk file FT03F001 for future reading by the PLOT program. The modified program is available from the author.

III. Code Reconciliation Program - RECON

A. Introduction

In a major programming effort (such as the STAR model) where several people are independently developing program modules to be added to the model, the problem of bringing together the developing modules into a common master code is time consuming and liable to oversight errors. A code reconciliation program named RECON has been written to simplify this task. It presumes that two programs (call them A and B) have evolved from some common ancestor code. A and B are thus mostly identical, but involve changes to some subroutines of the model, the addition of some new routines, and possibly also deletion of some routines. The code reconciliation program reads the source code for the A and B programs and prints out information about routines that match, lines of code which differ, and routines that are present in one of A or B but absent from the other.

The RECON program is written in SIMSCRIPT. It is a fairly simple program as word-processors go, but is too complex to describe in detail here. The source code is available from the author. In this report we concentrate on the features, use, and limitations of the RECON program.

B. Inputs

The main inputs to the RECON program are the two programs, A and B, to be compared. Program A is read from SIMSCRIPT input unit 1 and program B is read from unit 2. The user can define these units to point to whatever physical devices are desired - usually to disk data sets containing the source program card images.

For example, the following IBM/360 JCL points unit 1 to a program A

stored in three source program modules on a disk. The three modules are automatically concatenated in the order given.

```
//GO.SIMU01 DD UNIT=2314,VOL=SER=DISK03,DISP=SHR,DSN=PROGA1
//           DD UNIT=2314,VOL=SER=DISK03,DISP=SHR,DSN=PROGA2
//           DD UNIT=2314,VOL=SER=DISK03,DISP=SHR,DSN=PROGA3
```

In addition, there are 3 inputs which are read from unit 5 -- usually 3 cards. The first card contains an integer input value FILE.LGTH in free format. It defines the maximum number of card images from each program kept in RECON arrays at any given time. If RECON finds a mismatch in a given routine of the A and B programs, then it will look up to FILE.LGTH cards ahead in each routine trying to reestablish a match. A reasonable value for FILE.LGTH is 50, larger values may slow execution since up to FILE.LGTH**2 card comparisons may be made for each failure to match.

The second input card contains an alphabetic description of program A (for example, its name) which is printed on the RECON output for identification purposes. Up to 80 characters may be used. The third input card describes program B in an analogous fashion.

C. Module Definition and Sequencing

RECON breaks each of programs A and B into a sequence of modules. A module is a segment of code which starts with a header card and ends with the word END.

Header cards are cards on which the first non-blank characters are the keywords:

PREAMBLE
MAIN
EVENT
UPON
ROUTINE

corresponding to the standard SIMSCRIPT program elements. Each keyword must be followed by a blank(if there is room on the card for it). The EVENT, UPON, and ROUTINE header cards should also contain a program element name following the keyword. RECON will store up to 19-character names. A name must be followed by a blank column or by a left parenthesis if there is room for it on the card. The SIMSCRIPT keywords TO and FOR may appear between the header keyword and the name. Since RECON ignores TO and FOR in this context, these two words may not be used as module names.

The end of a module is signalled by any card that has the characters END separate from other characters(that is, immediately preceeded and followed by blanks if there is room on the card) before any '' comment indicators. END need not be the first word on the card.

RECON operates on one module at a time (in each of A and B). If the current module headers match (i.e. both PREAMBLE, both MAIN, both ROUTINE with matching name, or both EVENT or UPON with matching name) then RECON will go inside the module and compare its contents line by line. (See section D.). If the current module headers do not match, then RECON will print one of the modules as, for example,

"MODULE IN FILE A - NOT IN FILE B",

and advance to the next module header in that file to attempt a match again.

In order to determine which program file to advance if a match does not occur, RECON assumes that both programs contain modules in the following standardized order. First comes the PREAMBLE, followed by the MAIN program, and then all other EVENT, UPON, and ROUTINE modules arranged so that their names are in alphabetic order.

CAUTION: Alphabetic order is defined by the IBM SIMSCRIPT collating sequence which is

ABC → XYZ012 → 89 space dot

Note that the space (blank) comes after the alphanumeric characters but before the dot (period). This results in a slightly nonstandard alphabetization in which the following is a proper sequence of program element names:

RESET
RES2
RES
RES.MOVE

RECON uses the alphabetic order of names only when deciding which file to advance if headers do not match. If both A and B have the same module headers and names in the same sequence, then the alphabetic ordering will never be considered. But if A has modules named (in sequence) J, K, L, M, N while those of B are J, N, K, L, M, then RECON will

compare J's
declare module K in file A - not B
declare module L in file A - not B
declare module M in file A - not B

compare N's
declare module K in file B - not A
declare module L in file B - not A
declare module M in file B - not A

with the result being that many of the modules are not compared.

D. Comparisons Inside a Module

When RECON has located a pair of matching headers, it proceeds to do line by line comparisons of program A to program B within the module. It should be noted that only the header keyword and the name are used in matching

header cards. Other information such as given and yielding arguments may be different on the header cards.

Lines from A and B including the headers are then compared character by character. If all 80 characters match, the lines are not printed, but a message is printed indicating the match:

"Matching block of 5 lines, A lines 16 to 20. B lines 17 to 21"

If the entire modules match, there will be only one such message for each module.

If any character of the current line in A fails to match the corresponding character of the current line in B, then the lines do not match. In this case RECON looks ahead in both files trying to find the closest place where a match can be reestablished. Up to FILE.LGTH future lines are considered in each of A and B (but never beyond the END of the current module).

Messages identifying the non-matching portions are printed such as:

```
BLOCK IN FILE A -- NOT IN FILE B
2 LINES, FROM LINE 24 TO LINE 25
24 LET X = 1
25 CALL TOTAL
```

All non-matching lines are printed for easy cross reference to programs A and B.

If a match cannot be reestablished by the end of the modules or within FILE.LGTH lines, then the module is abandoned - remaining lines are not compared, and both programs are advanced to the next module.

Any program lines which do not lie between header and end cards are identified as being orphans and are printed.

IV. Code Plot Program

As an aid for documentation of computer programs, a code plotting program has been written. Computer listings of computer programs are often too faint to reproduce well in program documentation and may require photo-reduction to conveniently fit technical report or thesis page sizes. The code plotting program uses the VERSATEC plotter to produce crisp easily reproduced page size copies of card decks. Line numbers may also be added. This program is written in FORTRAN and calls standard VERSATEC graphics subroutines [1]. A program listing (plotted by the program itself) appears in Figure 3 and includes JCL and sample data cards.

The code plot program assumes the following input: the information to be plotted is read from cards -- one card per plot line. All 80 card columns are plotted. The input deck of cards to be plotted is arranged into modules each of which starts a new plot page. Each module starts with a line number card (which is not plotted) and ends with a nines card (which is also not plotted.)

The line number card contains a single input integer N in I5 format. If $N \geq 0$, line numbers will be plotted at the left of each line of information. The first line number is N and successive line numbers increase by one for each line. If $N = -1$, the line number sequence continues uninterrupted from the previous module. If $N = -2$, no line numbers will be plotted.

The nines card signals the end of a module and terminates plotting for the current page. It contains 9's in the first eight column positions. (Thus it is impossible to plot a card which starts with 8 nines.)

Up to 50 lines are plotted on each page. Modules which are longer than 50 lines automatically continue on additional pages with the line number sequence unbroken until a nines card is encountered.

```

1      STANDARD JOB CARD WITH TIME=1 GOES HERE
2      // EXEC FORTCLGN
3      //FORT.SYSIN DD *
4      C PLOT PROGRAM FOR 80 CHAR CARD IMAGES IN TECH REPORT FORMAT
5          DIMENSION XBOX(7), YBOX(7), ITEXT(20)
6          DATA XBOX/0.0,0.0,0.85,0.5,0.0,0.0,1.0/,
7              - YBOX/0.0,11.0,:1.0,0.0,0.0,0.0,1.0/, ICHK/'9999'/
8          CALL PLOTS(0,0,0)
9          XST = 0.0
10         ZLINE = 1.0
11         C MAIN LOOP START NEW CODE SEGMENT
12         C GET LINE NUMBER CODE
13         50      READ (5,127,END=300) N
14         127      FORMAT(1S)
15         IF (N.GE.0) ZLINE = N
16         C MOVE OVER FOR NEW PAGE
17         100      XST = XST + 9.0
18         DO 120 I = 1,5
19         120      XBOX(I) = XBOX(I) + 9.0
20         C OUTLINE THE PAGE
21         CALL NEWPEN(1)
22         CALL LINE(XBOX,YBOX,5,1.0,0)
23         C PRINT THE PAGE
24         CALL NEWPEN(2)
25         X = XST + 2.0
26         XX = XST + 1.5
27         Y = 9.5
28         DO 200 I = 1,50
29         200      READ(5,157,END=300) ITEXT
30         157      FORMAT(20A4)
31         IF ((ITEXT(1).EQ.ICHK).AND. (ITEXT(2).EQ.ICHK)) GO TO 50
32         Y = Y - 0.15
33         CALL SYMBOL(X,Y,0.070,ITEXT,0.0,80)
34         IF (N .EQ. -2) GO TO 200
35         CALL NUMBER(XX,Y,0.07,ZLINE,0.0,-1)
36         ZLINE = ZLINE + 1
37         200      CONTINUE
38         GO TO 100
39         300      CALL PLOT (0.,0.,999)
40         STOP
41         END
42         REPLACE THIS CARD WITH A STANDARD ORANGE END OF FILE CARD
43         //GO.PLOTparm DD *
44         4PLOT XMIN=8.0,XMAX=150.0,YMIN=-3.0,YMAX=12.0 4END
45         //GO.SYSIN DD DATA
46         LINE NUMBER CARD GOES HERE
47         INPUT DECK TO BE PLOTTED GOES HERE
48         NINES CARD GOES HERE
49         REPLACE THIS CARD WITH A STANDARD ORANGE END OF FILE CARD

```

Figure 3. Code Plot Program

The last module in the input deck is ended with a nines card and followed by the standard orange end of file card /*.

REFERENCES

[1] "VERSATEC GRAPHICS PLOTTING MANUAL", Naval Postgraduate School,
W.R. Church Computer Center, Technical Note No. 0141-34, Feb. 1978.

INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2. Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
3. Department Chairman, Code 55 Department of Operations Research Naval Postgraduate School Monterey, California 93940	1
4. Professor James K. Hartman Code 55Hh Department of Operations Research Naval Postgraduate School Monterey, California 93940	40
5. Professor S. H. Parry, Code 55Py Department of Operations Research Naval Postgraduate School Monterey, California 93940	1
6. LTC Edward P. Kelleher, Code 55Ka Department of Operations Research Naval Postgraduate School Monterey, California 93940	1
7. Professor Arthur L. Schoenstadt, Code 53Zh Department of Mathematics Naval Postgraduate School Monterey, California 93940	1
8. Office of the Commanding General U.S. Army TRADOC Attn: General Donn A. Starry Ft. Monroe, Virginia 23651	1
9. Headquarters U.S. Army Training & Doctrine Command Attn: ATCG-T (Col. Ed Scribner) Ft. Monroe, Virginia 23651	1

10. Headquarters 1
U.S. Army Training & Doctrine Command
Attn: Director, Analysis Directorate
Combat Developments (MAJ Chris Needels)
Ft. Monroe, Virginia 23651

11. R. Stampfel, Code 55 1
Department of Operations Research
Naval Postgraduate School
Monterey, California 93940

END

DATE

FILMED

8-80

DTIC

